Elliptic Curves and Elliptic Curve Cryptography

An Honors Thesis (HONRS 499)

by

Amiee L. Rodal

Thesis Advisor

Michael A. Karls

Ball State University

Muncie, Indiana

May 2004

Date of Graduation

May 2004

Abstract

The purpose of this thesis is to introduce elliptic curves and their properties in order to see how they can be used to form groups over different types of fields. These groups are then applied to a cryptographic scheme known as Elliptic Curve Cryptography, and the encryption and decryption processes are demonstrated through the use of examples. Topics from abstract algebra, such as groups, abelian groups, and fields are discussed, as well as some background information about cryptography. Finally, the processes of addition and the encryption and decryption schemes will be implemented using *Mathematica* code originated from the equations presented in the literature.

Acknowledgements

This thesis would not have been possible without the help and support of several people. I would first like to thank my thesis advisor, Dr. Michael Karls, for his time, energy, and guidance throughout the year as we explored this topic together. I would also like to thank Chris O'Maley for his encouragement, and my family and friends for their support along the way.

Table of Contents

1.0 Introduction.....1
2.0 Elliptic Curves over the Real Numbers.....2
3.0 Groups, Abelian Groups, and Fields.....3
3.1 Groups.....4
3.2 Abelian Groups.....4
3.3 Fields.....5
4.0 Adding Points on Elliptic Curves over the Real Numbers.....6
4.1 The Graphical Approach.....6
4.1.1 Adding Distinct Points P and Q when P is not equal to –Q.....7
4.1.2 Adding the Points P and –P.....7
4.1.3 Doubling the Point P.....8
4.2 The Algebraic Approach.....9
4.2.1 Adding Distinct Points P and Q when P is not equal to –Q.....9
4.2.2 Adding the Points P and –P.....11
4.2.3 Doubling the Point P.....11
4.3 Adding Points on Elliptic Curves over Z_p.....11
4.3.1 Adding Distinct Points P and Q when P is not equal to –Q.....12
4.3.2 Adding the points P and –P.....12
4.3.3 Doubling the Point P.....12
5.0 Elliptic Curve Cryptography.....13
5.1 Elliptic Curve Cryptography Scheme, using Alice and Bob.....13
5.1.1 The Encryption Operation.....14
5.1.2 The Decryption Operation.....14
5.2 An Example of the Encryption and Decryption Operations.....14
6.0 Elliptic Curves over F_2^m.....15
6.1 Elliptic Curve Groups over F_2^m.....15
6.2 Adding Distinct Points P and Q when P is not equal to –Q.....15
6.3 Doubling the Point P.....17
6.4 ECC Scheme Using Elliptic Curve Groups Over F_2^m.....18
6.4.1 An Example of the Encryption and Decryption Operations.....18
References.....20

Elliptic Curves and Elliptic Curve Cryptography

1.0 Introduction

Quadratic equations are studied extensively within mathematics throughout a student's high school and college careers. The standard form for these equations (in the variable x) is given by

$$ax^2 + bx + c = 0,$$

where a, b, and c are real. The *quadratic formula*, given by

$$x = (-b \pm \sqrt{(b^2 - 4ac)}) / (2a),$$

is introduced in algebra. *Ellipses*, *parabolas*, and *hyperbolas* are studied in geometry, and surfaces such as *hyperboloids*, given by

$$(x / a)^2 + (y / b)^2 - (z / c)^2 = 1,$$

and *paraboloids*, given by

$$(x / a)^2 + (y / b)^2 = z,$$

are the focus in multivariable calculus [1]. All of these are quadratic equations. What is beyond quadratics? The answer is *elliptic curves*, which are curves of the form

$$y^2 = x^3 + ax^2 + bx + c.$$

The study of elliptic curves can be traced back to the ancient Greeks and Alexandrians, from which a deep theory has emerged. Their name comes from the work of G. C. Fagnano (1682-1766) who showed that computing the arc length of an ellipse leads to the integral

$$\int(1 / \sqrt{((1-u^2)(1-k^2u^2))})du.$$

By making the changes of variables,

$$v^2 = (1-u^2)(1-k^2u^2) = (u-\alpha)(u-\beta)(u-\gamma)(u-\delta)$$

followed by

$$x = 1/(u-\alpha) \text{ and } y = v/(u-\alpha)^2,$$

one is lead to the integral

$$\int(1 / \sqrt{(x^3 + ax^2 + bx + c)})dx,$$

which is why a curve of the form $y^2 = x^3 + ax^2 + bx + c$ is called an elliptic curve [1].

Elliptic curves have been used to study or solve famous problems, such as the Congruent Number Problem and Fermat's Last Theorem. A rational number n is said to be *congruent* if there exists a right triangle with rational sides whose area is n. For example, 6 is a congruent number, since the right triangle with sides 3, 4, and 5 has area 6. Mathematicians such as Pierre de Fermat (1601-1665) and Leonhard Euler (1707-1783) studied this problem which can be turned into an investigation of points on certain elliptic curves. Fermat's Last Theorem, which states that there are no non-zero integer solutions x, y, z to the equation $x^n + y^n = z^n$ for integer $n>2$, was proved in 1993 by Andrew Wiles. A key to Wiles' proof was to show that if Fermat's Last Theorem were false, a certain type of elliptic curve would exist that leads to a contradiction [1].

Elliptic curves can also be used as schemes to transmit information securely. In 1985, Neal Koblitz, from the University of Washington, and Victor Miller, who worked at IBM, first proposed the application of elliptic curve systems to *cryptography*, which is the science of concealing the meaning of a message [7], [11]. To *encrypt* a message, one conceals the meaning of the message using a code or cipher, and to *decrypt* the message, one turns the encrypted message back into the original message [11].

Many cryptosystems necessitate the use of an algebraic structure known as a group, and elliptic curves can be used to form such a structure, referred to as an elliptic curve group [7]. To understand elliptic curve groups, a good starting point is to look at elliptic curves over the real numbers. The next step is to consider elliptic curves over finite fields such as the integers modulo p, where p is a prime number or finite fields of polynomials.

These properties of elliptic curves and elliptic curve groups can then be applied to cryptographic schemes, known as *elliptic curve cryptography* (ECC) schemes. We will look at one such ECC scheme, known as the Elliptic Curve ElGamal Method. Elliptic curve cryptography, which is being implemented by users today, maintains the three objectives of information security: *confidentiality*, the concealment of data from unauthorized parties, *integrity*, the assurance that data is genuine, and *availability*, the fact that the system still functions efficiently after security provisions are in place [2]. Elliptic curve cryptography has expanded the use of public-key cryptosystems, providing systems of encryption that are easier to implement and harder to crack [9].

The graphs and tables for the examples in this paper were generated with the *computer algebra system* Mathematica. Sample Mathematica code which can be modified to create other examples is provided in the Appendices.

2.0 Elliptic Curves over the Real Numbers

An *elliptic curve over real numbers* is the set of points (x, y) that satisfy an equation of the form:

$$y^2 = x^3 + ax + b \qquad (1)$$

where x, y, a, and b are real numbers [7]. There are other elliptic curves of the more general "Weierstrass" form:

$$y^2 + a_1xy + a_2y = a_3x^3 + a_4x^2 + a_5x + a_6,$$

but through a change of variable, one can put any elliptic curve over the reals into form (1) [6], [9]. Figure 1 shows some examples of elliptic curves.

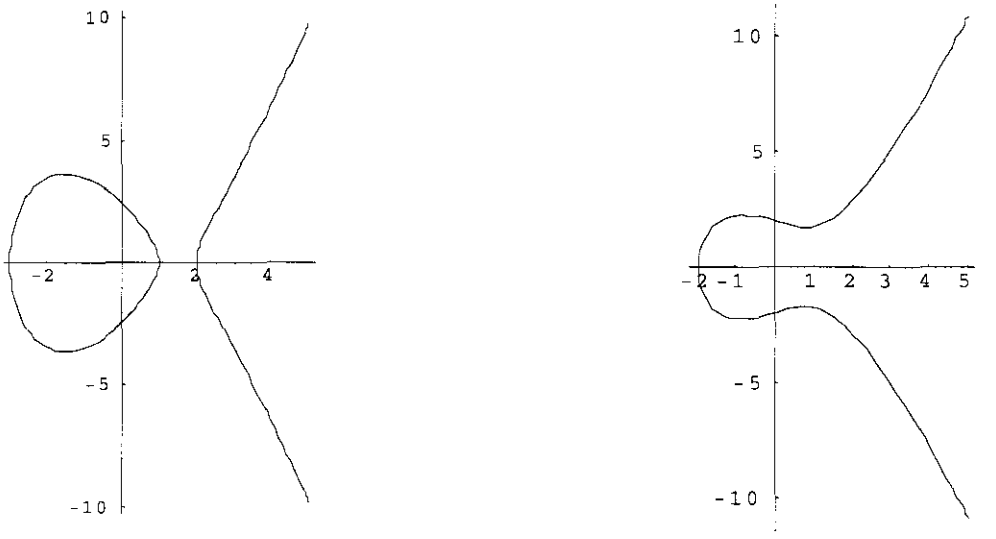


Figure 1: Elliptic curves $y^2 = x^3 - 7x + 6$ on the left and $y^2 = x^3 - 2x + 4$ on the right.

Elliptic curves of type (1) can be divided into two groups, *non-singular* and *singular* elliptic curves. A continuously differentiable curve written in the form

$$F(x, y) = 0$$

is said to be non-singular if there are no points on the curve at which both partial derivatives of F are zero. For such curves, it follows from the Implicit Function Theorem that at every point on the curve, there will be a tangent line [10]. It can be shown that any elliptic curve for which the right-hand side has three distinct roots will be non-singular, and a necessary and sufficient condition for the cubic polynomial $x^3 + ax + b$ to have three distinct roots is that $4a^3 + 27b^2$ is not equal to zero [6]. We will only use non-singular curves, as we will need to have curves at which each point has a tangent line. The two curves pictured in Figure 1 are both non-singular, as $4(-7)^3 + 27(6)^2 = -400$ and $4(-2)^3 + 27(4)^2 = 400$.

3.0 Groups, Abelian Groups, and Fields

The subject of Abstract Algebra plays a strong role in many branches of mathematics, including cryptography. Most of the ideas discussed in this section can be found in [4].

3.1 Groups

Algebraic structures known as groups arise naturally in the study of symmetry, geometric transformations, algebraic coding theory, and in the analysis of the roots of polynomial equations. Many cryptosystems, including elliptic curve cryptography, require the use of algebraic groups. A *group* is a nonempty set G with a binary operation, denoted with *, that satisfies the following axioms:

1. *Closure*: If a and b are in G, then a*b is in G
2. *Associativity*: a*(b*c) = (a*b)*c for all a, b, c in G
3. There is an element e in G, called the *identity* element, such that a*e = a = e*a for every a in G
4. For each a in G, there is an element d in G, called the *inverse* of a, such that a*d = e and d*a = e.

One should note that a group is said to be of *finite order* if it has a finite number of elements.

3.2 Abelian Groups

An *abelian*, or commutative, group is a special type of group, which satisfies one further axiom:

5. *Commutativity*: a*b = b*a for all a and b in G.

An example of an abelian group is the *integers modulo n*, denoted Z_n. For positive integer n, Z_n consists of the integers {0, 1, ..., n-1}. To sum a and b in Z_n, we add a to b and take the remainder on division by n. As an example, here is the addition table for Z_{11}.

Example 1: Addition table for Z_{11}

+	0	1	2	3	4	5	6	7	8	9	10
0	0	1	2	3	4	5	6	7	8	9	10
1	1	2	3	4	5	6	7	8	9	10	0
2	2	3	4	5	6	7	8	9	10	0	1
3	3	4	5	6	7	8	9	10	0	1	2
4	4	5	6	7	8	9	10	0	1	2	3
5	5	6	7	8	9	10	0	1	2	3	4
6	6	7	8	9	10	0	1	2	3	4	5
7	7	8	9	10	0	1	2	3	4	5	6
8	8	9	10	0	1	2	3	4	5	6	7
9	9	10	0	1	2	3	4	5	6	7	8
10	10	0	1	2	3	4	5	6	7	8	9

3.3 Fields

In order to work with elliptic curve cryptography, one also needs to become familiar with fields. A *field* is a nonempty set F equipped with two operations (usually written as addition and multiplication) that satisfy the following axioms:

For all a, b, c in F,

1. a+b is in F
2. a+(b+c) = (a+b)+c
3. a+b = b+a
4. There is an element 0_F in F such that $a+0_F = a = 0_F + a$
5. The equation $a + x = 0_F$ has a solution in F
6. ab is in F
7. a(bc) = (ab)c
8. a(b+c) = ab + ac and (a+b) c = ac + bc (Distributive Law)
9. ab = ba
10. There is an element 1_F in F such that $a1_F = a = 1_F$ for 1_F not equal to 0_F
11. For each a not equal to 0_F in F, the equation $ax = 1_F$ has a solution in F.

Some examples of fields include the set of real numbers, **R**, with the usual addition and multiplication, the set of complex numbers, **C** = {a+b*i* | a and b are real}, and Z_p, when p is prime. Multiplication works the same way as addition in Z_p. It is important to note that Z_p has a finite number of elements, thus making it a finite field. Here is the multiplication table for Z_{11}.

Example 2: Multiplication table for Z_{11}

*	0	1	2	3	4	5	6	7	8	9	10
0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7	8	9	10
2	0	2	4	6	8	10	1	3	5	7	9
3	0	3	6	9	1	4	7	10	2	5	8
4	0	4	8	1	5	9	2	6	10	3	7
5	0	5	10	4	9	3	8	2	7	1	6
6	0	6	1	7	2	8	3	9	4	10	5
7	0	7	3	10	6	2	9	5	1	8	4
8	0	8	5	2	10	7	4	1	9	6	3
9	0	9	7	5	3	1	10	8	6	4	2
10	0	10	9	8	7	6	5	4	3	2	1

Another example of a field is F_2^m, where m is a nonnegative integer [9]. In order to understand this example, we need to define some terms. A nonempty set R that satisfies field axioms 1-8 is called a *ring*. If R also satisfies axiom 9, we say R is a *commutative ring*. If ring R satisfies axiom 10, with $1_R = 0_R$ allowed, R is a *ring with identity*. An element *a* in a ring R with identity is called a *unit* if there exists a *u* in R such that $au = 1_R = ua$. An element *a* in a commutative ring with identity *R* is said to be an *associate* of an element *b* of *R* if $a = bu$ for some unit *u*.

A *polynomial with coefficients in ring* R is an expression of the form

$$p(x) = a_0 + a_1x + a_2x^2 + \ldots + a_nx^n$$

where n is a nonnegative integer and the coefficients a_i are elements of R. Note that the set of polynomials with coefficients in ring R, denoted R[x] is also a ring. If F is a field, a nonconstant polynomial p(x) in F[x] is said to be *irreducible* if its only divisors are its associates and the nonzero constant polynomials (units).

F_2^m is the representation for a field of polynomials modulo a given irreducible polynomial p(x), with coefficients in Z_2. This type of modular arithmetic is done in a similar manner as Z_p. To find the sum or product of polynomials f(x) and g(x) in F_2^m, simply add or multiply the polynomials, divide by p(x) using long division, and take the remainder, keeping in mind that all coefficients are either 0 or 1. For example, the field $F_2^3 = \{0, 1, x, x^2, x^2 + x, x^2 + 1, x + 1, x^2 + x + 1\}$ is the set of polynomials in $Z_2[x]$ modulo the polynomial $p(x) = x^3 + x + 1$ [12]. Note that by suppressing the powers of x, we get a string of 1's and 0's, so fields of this type can be used to represent bit strings of 0's and 1's. Computers can perform arithmetic in these fields very efficiently [9].

4.0 Adding Points on Elliptic Curves over the Real Numbers

A binary operation, usually denoted by addition, defined over a non-singular elliptic curve of form (1), E, can be used to transform the curve into an abelian group. An *elliptic curve group* over the real numbers consists of the points on the curve, along with a special point Ó, called the *point at infinity*, which will be the identity element under this addition operation.

The adding of points on elliptic curves can be done using two different methods, graphical and algebraic. The key to each approach is to find the third point of intersection of a line with an elliptic curve, given two of the points of intersection. Any vertical line will contain the point at infinity and tangent lines contain the point of tangency twice [6].

4.1 The Graphical Approach

Define the *negative of the point at infinity* to be $-Ó = Ó$ and the *negative* of any other point $P = (x_P, y_P)$ on elliptic curve E to be its reflection over the x-axis, that is $-P = (x_P, -y_P)$. Note that if $P = (x_P, y_P)$ is on the curve (1), then so is –P. The graphical approach is broken into three cases:

1. Adding two distinct points P and Q with P not equal to -Q

2. Adding the points P and –P
3. Doubling the point P (i.e. adding the point P to itself)

4.1.1 Adding Distinct Points P and Q when P is not equal to -Q

Suppose that P and Q are distinct points on an elliptic curve with P not equal to -Q. To add the point P to Q, a line is drawn between the two points and extended until it crosses the elliptic curve at a third point, –R. This point is then reflected over the x-axis to its negative R. The addition of points P and Q is defined to be: P + Q = R.
Figure 2 gives an example of this case.

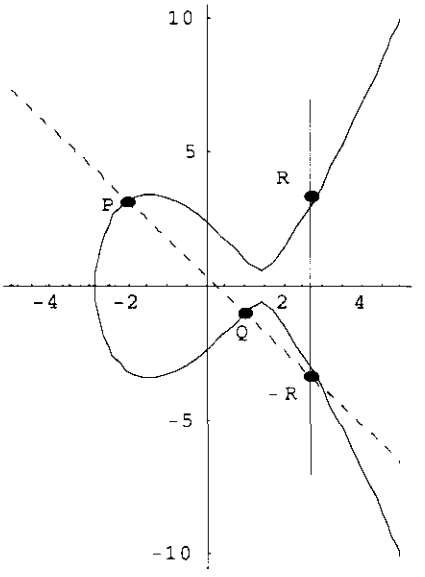


Figure 2: For P = (-2, 3.162) and Q = (1, -1) on $y^2 = x^3 - 6x + 6$, P + Q = R with R = (2.684, 3.336).

4.1.2 Adding the Points P and –P

The adding of the points P and –P poses a unique situation. The line through the two points is a vertical line, which will not intersect the elliptic curve at a third point, so we define, P + (-P) = O, the point at infinity. Figure 3 gives an example of this case.

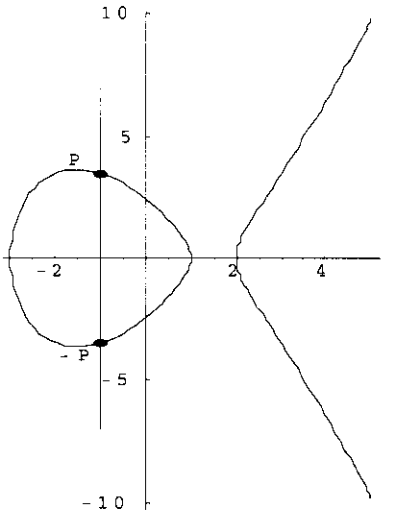


Figure 3: For P = (-1, 3.464) and -P = (-1, -3.464) on $y^2 = x^3 - 7x + 6$, P + (-P) = Ó.

4.1.3 Doubling the Point P

The doubling of a point P poses yet another unique situation. Instead of drawing a line between two different points, the tangent line to the curve at the point P is drawn and extended until it crosses the elliptic curve at one other point, called –R. If the y-coordinate of P is zero, this tangent line will be vertical, so we are in the second case. Otherwise, as in the first case, point –R is reflected over the x-axis to its negative, R. Thus, the doubling of the point P is defined to be: 2P = P + P = R. Figures 4 and 5 illustrate this case.

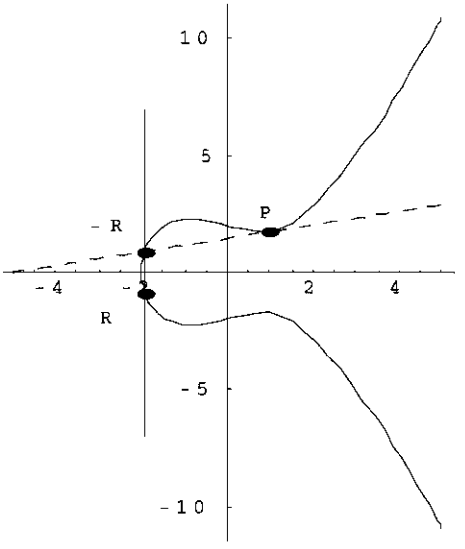


Figure 4: Doubling P = (1, 1.732) on the curve $y^2 = x^3 - 2x + 4$.

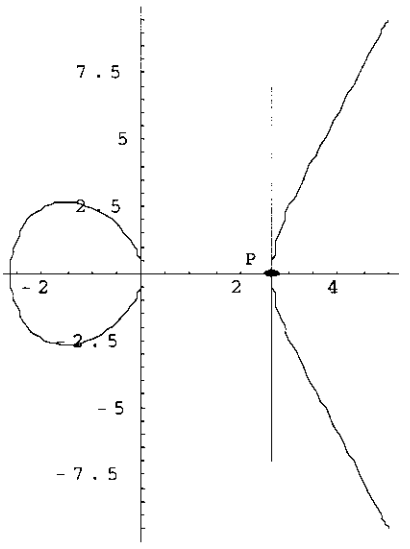


Figure 5: Doubling the point P = (2.646, 0) on the curve $y^2 = x^3 - 7x$.

4.2 The Algebraic Approach

The graphical approach provides an excellent method of illustrating elliptic curve addition. However, it is not a practical method of implementing arithmetic computations because one has to either estimate the coordinates of the points that are being added, or solve different sets of equations to find the exact coordinates of each of the points. For these reasons, a more efficient approach to add points on elliptic curves is to use specific algebraic formulas for the addition. This method also makes the definition of elliptic curve addition more rigorous.

As in the graphical case, define the point at infinity to be the additive identity, the negative of the point at infinity to be $-\mathcal{O} = \mathcal{O}$, and the negative of any other point P on elliptic curve (1) to be its reflection about the x-axis. Again, there are three additional cases we need to consider for adding points on elliptic curves:

1. Adding distinct points P and Q, when P is not equal to -Q
2. Adding the points P and –P
3. Doubling the point P.

4.2.1 Adding Distinct Points P and Q when P is not equal to -Q

Suppose $P = (x_P, y_P)$ and $Q = (x_Q, y_Q)$ and that P is not equal to –Q. Then P + Q = R where $-R = (x_R, y_{(-R)}) = (x_R, -y_R)$ is the third point of intersection of the line through P and Q with the elliptic curve. In this case, $R = (x_R, y_R)$ is given by

$s = (y_P - y_Q) / (x_P - x_Q)$
$x_R = s^2 - x_P - x_Q$ and
$y_R = -y_P + s(x_P - x_R)$ [7].

Note: s is just the slope of the line through the points P and Q.

The x coordinate of R can be solved for by using the equation of the line through the two points P and Q, the given elliptic curve equation (1), and Vieta's formula which will be given below. The equation of the line through P and Q is

$$y - y_P = s(x - x_P). \tag{2}$$

Solving (2) for y and substituting for y in equation (1), we find

$$(s(x - x_P) + y_P)^2 = x^3 + ax + b. \tag{3}$$

Expanding equation (3),

$$s^2(x - x_P)^2 + 2s(x - x_P)y_P + y_P^2 = x^3 + ax + b,$$

$$s^2(x^2 - 2xx_P + x_P^2) + 2sxy_P - 2sx_Py_P + y_P^2 = x^3 + ax + b. \tag{4}$$

Moving the left-hand side (LHS) of (4) to the right-hand side (RHS) of equation (4) yields

$$0 = x^3 - s^2x^2 + 2s^2xx_P - s^2x_P^2 - 2sxy_P + 2sx_Py_P + ax - y_P^2 + b.$$

Using Vieta's formula[1], it can be seen that the coefficient of the x^2 term will be the negative of the sum of the three roots [5]. Thus,

$$s^2 = x_P + x_Q + x_R. \tag{5}$$

Solving equation (5) for x_R gives

$$x_R = s^2 - x_P - x_Q.$$

The y coordinate of R is easily found by substituting x_R into equation (2) and reflecting the point –R over the x-axis. Putting x_R into equation (2),

$$y_{(-R)} = y_P + s(x_R - x_P).$$

Reflecting $-R = (x_R, y_{(-R)}) = (x_R, -y_R)$ over the x-axis just changes the sign of the y-coordinate, yielding:

$$y_R = -y_P - s(x_R - x_P) = -y_P + s(x_P - x_R).$$

[1] Vieta's Formula (1540-1603): $a_3x^3 + a_2x^2 + a_1x + a_0 = a_3(x - r_1)(x - r_2)(x - r_3)$
$= a_3[x^3 - x^2(r_1 + r_2 + r_3) + x(r_1r_2 + r_2r_3 + r_1r_3) - r_1r_2r_3]$ [8]

4.2.2 Adding the points P and -P

The line through P and –P is a vertical line, so we define their sum to be Ó, the point at infinity, as in the graphical approach. Thus, P + (-P) = Ó.

4.2.3 Doubling the Point P

If the y-coordinate of point P = (x_P, y_P) is zero, we are in Case 2. When y_P is not 0, we take 2P = P + P = R where R = (x_R, y_R) is given by:

$s = (3x_P^2 + a) / 2y_P$
$x_R = s^2 - 2x_P$ and
$y_R = -y_P + s(x_P - x_R)$ [7].

Note that s is the slope of the line tangent to the curve at the point P. The equation $s = (3x_P^2 + a) / 2y_P$ can be found by implicitly differentiating the elliptic curve equation $y^2 = x^3 + ax + b$. Thus,

$$\begin{aligned} y^2 &= x^3 + ax + b \\ 2yy' &= 3x^2 + a \\ y' &= (3x^2 + a) / 2y, \end{aligned}$$

and the result follows by substituting in the coordinates of P. The x_R and y_R coordinates of R are found using the same method as for Case 1. However, because $x_P = x_Q$ in this case,

$$x_R = s^2 - 2x_P \quad \text{and} \quad y_R = -y_P + s(x_P - x_R).$$

With our definition for addition on non-singular elliptic curves, it should be clear that the group properties of closure, and commutativity are upheld. The set has an identity element, which is the point at infinity, and every point has an inverse, as P + (-P) = Ó. The axiom of associativity is not as clear and is difficult to prove, but is sustained under this operation none-the-less [3]. Thus, an abelian group is formed.

4.3 Adding Points on Elliptic Curves over Z_p

The addition of points on elliptic curves over the real numbers is a good approach to see the underlying steps in performing the operation. However, calculations prove to be slow and inaccurate due to rounding errors, and the implementation of these calculations into cryptographic schemes requires fast and precise arithmetic [7]. Therefore elliptic curve groups over finite fields such as Z_p, when p>3 is prime, are used in practice [6].

An elliptic curve with Z_p as its underlying field can be formed by choosing a and b within the field Z_p. Similar to the real case, the curve includes all points (x, y) that satisfy the elliptic curve equation

$$y^2 \pmod p = (x^3 + ax + b) \pmod p,$$

where x and y are numbers in Z_p [7]. Note that there are only finitely many points on this type of curve.

As in the real case, if $(4a^3 + 27b^2) \pmod p$ is not 0, then the corresponding elliptic curve forms a group [6]. This group consists of the points on the curve, along with Ó, the point at infinity. Again, we define the negative of the point at infinity to be - Ó = Ó and the negative of a point $P = (x_P, y_P)$ to be $-P = (x_P, (-y_P) \pmod p)$.

The arithmetic in an elliptic curve group over Z_p is very similar to that done algebraically with elliptic curve groups over the real numbers- the only difference is that all calculations are performed (mod p) [7].

4.3.1 Adding Distinct Points P and Q when P is not equal to –Q

Suppose $P = (x_P, y_P)$ and $Q = (x_Q, y_Q)$ and that P is not equal to –Q. Then $P + Q = R$ where:

$s = ((y_P - y_Q)(x_P - x_Q)^{-1}) \pmod p$
$x_R = (s^2 - x_P - x_Q) \pmod p$ and
$y_R = (-y_P + s(x_P - x_R)) \pmod p$ [7].

4.3.2 Adding the points P and –P

The line through P and –P is a vertical line, so we define their sum to be Ó, the point at infinity, as before. Thus, P + (-P) = Ó.

4.3.3 Doubling the Point P

If the y-coordinate of P is zero (mod p), then we are in the case where P = -P. To double the point $P = (x_P, y_P)$ where y_P is not 0, we take $2P = P + P = R$ where:

$s = ((3x_P^2 + a)(2y_P)^{-1}) \pmod p$
$x_R = (s^2 - 2x_P) \pmod p$ and
$y_R = (-y_P + s(x_P - x_R)) \pmod p$ [7].

Example 3: The addition table for the points on the elliptic curve $y^2 = x^3 + 5x + 4$ over Z_{11}.

*	{0, 2}	{0, 9}	{2, 0}	{4, 0}	{5, 0}	{10, 3}	{10, 8}	∞
{0, 2}	{5, 0}	∞	{10, 8}	{10, 3}	{0, 9}	{2, 0}	{4, 0}	{0, 2}
{0, 9}	∞	{5, 0}	{10, 3}	{10, 8}	{0, 2}	{4, 0}	{2, 0}	{0, 9}
{2, 0}	{10, 8}	{10, 3}	∞	{5, 0}	{4, 0}	{0, 9}	{0, 2}	{2, 0}
{4, 0}	{10, 3}	{10, 8}	{5, 0}	∞	{2, 0}	{0, 2}	{0, 9}	{4, 0}
{5, 0}	{0, 9}	{0, 2}	{4, 0}	{2, 0}	∞	{10, 8}	{10, 3}	{5, 0}
{10, 3}	{2, 0}	{4, 0}	{0, 9}	{0, 2}	{10, 8}	{5, 0}	∞	{10, 3}
{10, 8}	{4, 0}	{2, 0}	{0, 2}	{0, 9}	{10, 3}	∞	{5, 0}	{10, 8}
∞	{0, 2}	{0, 9}	{2, 0}	{4, 0}	{5, 0}	{10, 3}	{10, 8}	∞

5.0 Elliptic Curve Cryptography

Having defined the addition of points on elliptic curves over Z_p, we now look at how to apply these ideas to an ECC scheme known as the ElGamal scheme.

5.1 Elliptic Curve Cryptography Scheme, using Alice and Bob

ECC schemes are a form of public-key cryptosystems. Public-key cryptosystems are a relatively new technology, developed in 1976 by Whitfield Diffie and Martin Helman, both Stanford researchers. These cryptosystems involve separate encryption and decryption operations. The encryption rule uses a *public key*, while the decryption rule employs a *private key*. Knowledge of the public key allows encryption of a message but does not permit decryption of the encrypted message. The private key is kept secret so that only the intended individual can decrypt the message [2].

ECC schemes use an elliptic curve E over finite fields such as Z_p, where $p>3$ is prime and involve both an encryption and decryption operation. There are several public key schemes that can be used to encrypt and decrypt messages, such as the Diffie-Hellman scheme, the Vanstone-Menezes scheme, and the ElGamal scheme. We will look at the ElGamal encryption and decryption scheme. For more on the ElGamal scheme in general or how the other schemes work, see [6], [9], or [12].

The ElGamal public-key cryptosystem is based on the Discrete Logarithm problem in Z_p^*, the set of integers {1, 2, ..., p-1}, under multiplication modulo p. The utility of the Discrete Logarithm problem in a cryptographic setting is that finding discrete logarithms is difficult, but the inverse operation of exponentiation can be computed efficiently [12]. In other words, if a person is given α, β, and $\alpha^a = \beta \pmod p$, then it is very difficult to figure out the exponent a. We will use this idea in an ECC cryptosystem and perform the operations on an elliptic curve over Z_p. Note that in an elliptic curve group, α^a is interpreted as adding α to itself a times. This scheme will be demonstrated using the standard convention of Alice and Bob as sender and receiver of the message, respectively.

5.1.1 The Encryption Operation

Step 1: Bob chooses a point α on the elliptic curve E over Z_p and a number z.

Step 2: Bob computes $\beta = z\,\alpha$, and publishes α, β, E, and p.
He keeps his private key z secret.

Step 3: Suppose Alice wants to send a message to Bob. Alice picks an integer k, $1 < k <$ order of E, which will be her private key.

Step 4: To encrypt a message, Alice looks up Bob's public key. She then converts the message into points on the elliptic curve E.

Next, Alice performs the encryption operation:

$$e_k(x, k) = (k\alpha, x + k\beta)$$
$$= (y_1, y_2)$$

to encrypt the message [12]. Thus, the encrypted message is $y = (y_1, y_2)$.

5.1.2 The Decryption Operation

Step 5: Alice sends Bob the encrypted message. To decrypt the message, Bob uses the decryption operation:
$d_k(y_1, y_2) = y_2 - zy_1 = x + k\beta - zk\alpha = x + zk\alpha - zk\alpha = x$

where z is Bob's private key [12].

5.2 An example of the Encryption and Decryption Operations

Step 1: E is the elliptic curve $y^2 = x^3 + 5x + 4$ over Z_{11}
$\alpha = (10, 3)$
$z = 3$

Step 2: $\beta = 3(10, 3) = (10, 8)$
Bob's public key: $\alpha = (10, 3)$, $\beta = (10, 8)$,
E is $y^2 = x^3 + 5x + 4$ over Z_{11}
Bob's private key: $z = 3$

Step 3: Alice chooses $k = 2$

Step 4: Bob's public key: $\alpha = (10, 3)$, $\beta = (10, 8)$,
E is $y^2 = x^3 + 5x + 4$ over Z_{11}
Alice's message is $x = (2, 0)$, which is a point on the elliptic curve E.

$$y_1 = 2(10, 3) = (5, 0)$$

$y_2 = (2, 0) + 2(10, 8)$
$= (2, 0) + (5, 0)$
$= (4, 0)$
The encrypted message is $y = ((5, 0), (4, 0))$.

Step 5: $y = ((5, 0), (4, 0))$
$x = (4, 0) - 3(5, 0)$
$= (4, 0) - (5, 0)$
$= (4, 0) + (5, 0)$
$= (2, 0)$
The decrypted message is $x = (2, 0)$.

Appendix 2 shows this example performed in Mathematica.

6.0 Elliptic Curves over F_2^m

The entire process of adding points on elliptic curves over fields to form groups and encrypt information can be done with elliptic curves over F_2^m.

6.1 Elliptic Curve Groups over F_2^m

An *elliptic curve with an underlying field of polynomials, F_2^m* is formed by choosing a and b within the underlying field, as long as b is not equal to zero. The form of the elliptic curve equation for this case is

$$y^2 + xy = x^3 + ax^2 + b.$$

It includes all of the points (x, y) that satisfy the elliptic curve equation where x and y are elements of F_2^m, along with the point at infinity. As with Z_p, addition and multiplication of elements in F_2^m is done modulo an irreducible polynomial p(x), with coefficients in Z_2. As before, the negative of the point at infinity is the point at infinity. In order to get a group, the *negative* of any other point $P = (x_P, y_P)$ on the curve is $-P = (x_P, x_P + y_P)$ [9]. Just like elliptic curves over Z_p, there are only finitely many points on this type of elliptic curve, and the two cases of P + (-P) and 2P when $x_P = 0$ are defined to be the point at infinity.

6.2 Adding Distinct Points P and Q when P is not equal to -Q

Suppose that $P = (x_P, y_P)$ and $Q = (x_Q, y_Q)$ and Q is not the negative of P. Then $P + Q = R$ where R is the negative of $-R = (x_R, y_{(-R)})$:

$s = (y_P - y_Q)(x_P + x_Q)^{-1}$
$x_R = s^2 + s + x_P + x_Q + a$ and
$y_R = s(x_P + x_R) + x_R + y_P$ [7].

The x_R coordinate of R can be solved for using the equation of the line through the two points P and Q, the given elliptic curve equation, and Vieta's formula. The equation of the line through P and Q is

$$y - y_P = s(x - x_P).$$

Solving for y gives

$$y = s(x - x_P) + y_P. \tag{6}$$

Recall that the elliptic curve equation in this case is

$$y^2 + xy = x^3 + ax^2 + b. \tag{7}$$

Substituting equation (6) into equation (7), we find

$$(s(x - x_P) + y_P)^2 + x(s(x - x_P) + y_P) = x^3 + ax^2 + b. \tag{8}$$

Expanding equation (8),

$$s^2(x^2 - 2xx_P + x_P^2) + 2sxy_P - 2sx_Py_P + y_P^2 + sx^2 - sxx_P + xy_P = x^3 + ax^2 + b. \tag{9}$$

Moving the LHS of (9) to the RHS of (9) yields

$$0 = x^3 + (a - s - s^2)x^2 + 2s^2xx_P - s^2x_P^2 - 2sxy_P + 2sx_Py_P - y_P^2 + sxx_P - xy_P + b.$$

Using Vieta's formula, it can be seen that the coefficient of the x^2 term will be the negative of the sum of the three roots [5]. Thus,

$$-a + s + s^2 = x_P + x_Q + x_R. \tag{10}$$

Solving equation (10) for x_R gives

$$x_R = s^2 + s - a - x_P - x_Q. \tag{11}$$

Because addition and subtraction are the same for elements in Z_2, equation (11) is the same as

$$x_R = s^2 + s + a + x_P + x_Q.$$

The y_R coordinate of R is easily found by simply substituting x_R into equation (6) to find the y-coordinate of –R and taking the negative of the point -R. Putting x_R into equation (6) gives

$$y_{(-R)} = s(x_R - x_P) + y_P.$$

Taking the negative of the point –R yields

$$y_R = y_{(-R)} + x_R = s(x_R - x_P) + y_P + x_R$$
$$= s(x_P + x_R) + x_R + y_P$$

6.3 Doubling the Point P

To double the point P when x_P is not 0, we take 2P = P + P = R where:

$s = x_P + (y_P)\,(x_P)^{-1}$
$x_R = s^2 + s + a$ and
$y_R = x_P^2 + (s + 1)x_R$ [9].

The equation $s = x_P + (y_P)\,(x_P)^{-1}$ can be found by implicitly differentiating the elliptic curve equation $y^2 + xy = x^3 + ax^2 + b$. Thus,

$$y^2 + xy = x^3 + ax^2 + b$$
$$2yy' + y + xy' = 3x^2 + 2ax$$
$$y'(2y + x) = 3x^2 + 2ax - y$$
$$y' = (3x^2 + 2ax - y)\,(2y + x)^{-1}$$

But because all coefficients are taken modulo 2, the equation becomes

$$y' = (x^2 - y)\,(x)^{-1}$$
$$= x + (y)\,(x)^{-1}.$$

Substituting in the coordinates of P, we get the slope, s, of the line tangent to the curve at the point P. The x_R and y_R coordinates are found using the method shown in Case 1. However, because $x_P = x_Q$ in this case,

$$x_R = s^2 + s + a.$$

The y_R coordinate of R is found by substituting x_R into equation (6) to find the y-coordinate of –R and taking the negative of the point -R. Putting x_R into equation (6) we find

$$y_{(-R)} = s(x_R - x_P) + y_P$$

Taking the negative of the point –R,

$$y_R = y_{(-R)} + x_R = s(x_R - x_P) + y_P + x_R$$
$$= s(x_P + x_R) + y_P + x_R$$
$$= (x_P + (y_P)(x_P)^{-1})(x_P + x_R) + y_P + x_R$$
$$= x_P^2 + y_P + x_P x_R + (y_P x_R)\,(x_P)^{-1} + y_P + x_R$$
$$= x_P^2 + (x_P + (y_P)(x_P)^{-1} + 1)x_R$$
$$= x_P^2 + (s + 1)x_R.$$

Example 4: The group table for the set of points on the elliptic curve $y^2 + xy = x^3 + (x + 1)x^2 + (x + 1)$ over F_2^3.

*	$\{0, 1+x+x^2\}$	$\{1+x, 1+x^2\}$	$\{1+x, x+x^2\}$	$\{x^2, x\}$	$\{x^2, x+x^2\}$	∞
$\{0, 1+x+x^2\}$	∞	$\{x^2, x\}$	$\{x^2, x+x^2\}$	$\{1+x, 1+x^2\}$	$\{1+x, x+x^2\}$	$\{0, 1+x+x^2\}$
$\{1+x, 1+x^2\}$	$\{x^2, x\}$	$\{1+x, x+x^2\}$	∞	$\{x^2, x+x^2\}$	$\{0, 1+x+x^2\}$	$\{1+x, 1+x^2\}$
$\{1+x, x+x^2\}$	$\{x^2, x+x^2\}$	∞	$\{1+x, 1+x^2\}$	$\{0, 1+x+x^2\}$	$\{x^2, x\}$	$\{1+x, x+x^2\}$
$\{x^2, x\}$	$\{1+x, 1+x^2\}$	$\{x^2, x+x^2\}$	$\{0, 1+x+x^2\}$	$\{1+x, x+x^2\}$	∞	$\{x^2, x\}$
$\{x^2, x+x^2\}$	$\{1+x, x+x^2\}$	$\{0, 1+x+x^2\}$	$\{x^2, x\}$	∞	$\{1+x, 1+x^2\}$	$\{x^2, x+x^2\}$
∞	$\{0, 1+x+x^2\}$	$\{1+x, 1+x^2\}$	$\{1+x, x+x^2\}$	$\{x^2, x\}$	$\{x^2, x+x^2\}$	∞

6.4 ECC Scheme Using Elliptic Curve Groups over F_2^m

The ElGamal ECC scheme discussed in section five can also be implemented with elliptic curve groups over F_2^m. The encryption and decryption operations given by

$$e_k(x, k) = (k\alpha, x + k\beta) = (y_1, y_2)$$

and

$$d_k(y_1, y_2) = y_2 - zy_1$$

remain the same, but we use the new addition operations for elliptic curve groups over F_2^m shown in sections 6.2 and 6.3.

6.4.1 An Example of the Encryption and Decryption Operations

Step 1: E is the elliptic curve $y^2 + xy = x^3 + (x + 1)x^2 + (x + 1)$ over F_2^3
$\alpha = (x^2, x)$
$z = 2$

Step 2: $\beta = 2(x^2, x) = (1+x, x + x^2)$
Bob's public key: $\alpha = (x^2, x)$, $\beta = (1+x, x + x^2)$,
E is $y^2 + xy = x^3 + (x + 1)x^2 + (x+1)$ over F_2^3
Bob's private key: $z = 2$

Step 3: Alice chooses $k = 4$

Step 4: Bob's public key: $\alpha = (x^2, x)$, $\beta = (1+x, x + x^2)$,
E is $y^2 + xy = x^3 + (x + 1)x^2 + (x + 1)$ over F_2^3

Alice's message is $x = (x^2, x)$, which is a point on the elliptic curve E.

$y_1 = 4(x^2, x) = (1+x, 1 + x^2)$
$y_2 = (x^2, x) + 4(1+x, x + x^2)$
$= (x^2, x) + (1+x, x + x^2)$
$= (0, 1 + x + x^2)$
The encrypted message is $y = ((1+x, 1 + x^2), (0, 1 + x + x^2))$.

Step 5: $y = ((1+x, 1 + x^2), (0, 1 + x + x^2))$
$x = (0, 1 + x + x^2) - 2(1+x, 1 + x^2)$
$= (0, 1 + x + x^2) - (1+x, x + x^2)$
$= (0, 1 + x + x^2) + (1+x, 1 + x^2)$
$= (1 + x + x^2, x)$
The decrypted message is $x = (x^2, x)$.

Appendix 3 shows this example performed in Mathematica.

References

[1] Brown, E. (2000). Three Fermat trails to elliptic curves. *The College Mathematics Journal, 31*(3), 162-172.

[2] (2000, July). *The elliptic curve cryptosystem: An introduction to information security*. Retrieved October 3, 2003 from Certicom Web site: http://www.certicom.com

[3] Hastad, J., & Strom, S. (n.d.). *Elliptic curves*. Retrieved April 18, 2004 from, Seminars in Theoretical Computer Science Web site: http://www.nada.kth.se/kurser/kth/2D1441/lecturenotes/elliptic.pdf

[4] Hungerford, T. W. (1997). *Abstract Algebra: An introduction* (2nd ed.). New York: Saunders College Publishing.

[5] Joyce, D. E. (1999). *Dave's schort course on complex numbers quadratic and cubic equations*. Retrieved April 18, 2004 from Clark University, Department of Mathematics and Computer Science Web site: http://www.clarku.edu/~djoyce/complex/cubic.html

[6] Koblitz, N. (1998). *Algebraic aspects of cryptography*. Germany: Springer-Verlage Berlin Heidelberg.

[7] *Online ECC tutorial*. Retrieved October 10, 2003 from Certicom Web site: http://www.certicom.com/resources/ecc_tutorial/ecc_tut_1_0.html

[8] *Quartic equation*. Retrieved April 18, 2004 from MathWorld-A Wolfram Web Resource Web site: http://mathworld.wolfram.com/QuarticEquation.html

[9] Rosing, M. (1999). *Implementing elliptic curve cryptography*. Greenwich: Manning Publications Co.

[10] Rudin, W. (1964). *Principles of mathematical analysis* (3rd ed.). New York: McGraw-Hill, Inc.

[11] Singh, S. (1999). *The code book: The science of secrecy from Ancient Egypt to quantum cryptography*. New York: Anchor Books.

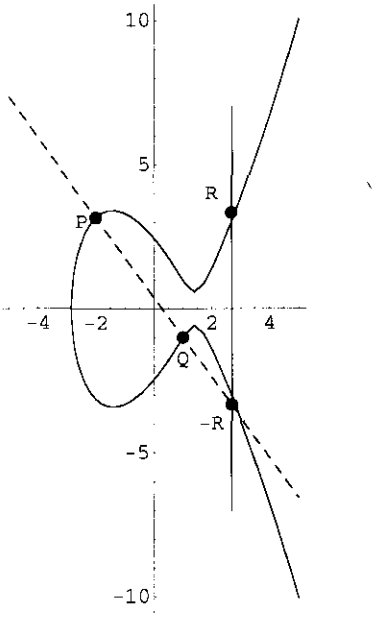
[12] Stinson, D. R. (2002). *Cryptography theory and practice* (2nd ed.). New York: Chapman & Hall/CRC.

Appendix 1

Sample *Mathematica* code for the graphs and tables in the paper.

Sample *Mathematica* code for creating graphs in the paper.

```
<< Graphics`ImplicitPlot`

<< Graphics`Graphics`

g0 = ImplicitPlot[{y^2 == x^3 - 6 x + 6, y == (Sqrt[10] + 1)/-3 (x - 1) - 1}, {x, -5, 5},
   PlotStyle → {GrayLevel[0], Dashing[{.03}]}, DisplayFunction → Identity];

g1 = Graphics[{RGBColor[1, 0, 1], Line[{{2.684, -7}, {2.684, 7}}]}];

g2 = TextListPlot[{{2, 4, "R"}, {2, -4, "-R"}, {-2.5, 3, "P"}, {1, -1.75, "Q"}},
   DisplayFunction → Identity];

g3 = Graphics[{PointSize[.04], RGBColor[0, 0, 1], Point[{2.684, -3.336}],
    Point[{2.684, 3.336}], Point[{-2, 3.162}], Point[{1, -1}]}];

Show[{g0, g1, g2, g3}, Axes → True,
  AspectRatio → Automatic, DisplayFunction → $DisplayFunction];
```



Sample *Mathematica* code for creating tables in the paper

```
T = Table[i, {i, 0, 10}];

A = Table[Mod[T[[k]] + T[[j]], 11], {k, 1, Length[T]}, {j, 1, Length[T]}];

J = TableForm[A, TableHeadings → {T, T}];

12 = Table[Insert[A[[i]], T[[i]], 1], {i, Length[A]}];

group1 = Insert[T, "+", 1];

13 = Insert[12, group1, 1];
```

```
DisplayForm[FrameBox[GridBox[13, RowLines → True, ColumnLines → True]]]
```

+	0	1	2	3	4	5	6	7	8	9	10
0	0	1	2	3	4	5	6	7	8	9	10
1	1	2	3	4	5	6	7	8	9	10	0
2	2	3	4	5	6	7	8	9	10	0	1
3	3	4	5	6	7	8	9	10	0	1	2
4	4	5	6	7	8	9	10	0	1	2	3
5	5	6	7	8	9	10	0	1	2	3	4
6	6	7	8	9	10	0	1	2	3	4	5
7	7	8	9	10	0	1	2	3	4	5	6
8	8	9	10	0	1	2	3	4	5	6	7
9	9	10	0	1	2	3	4	5	6	7	8
10	10	0	1	2	3	4	5	6	7	8	9

```
K = Table[i, {i, 0, 10}];

L = Table[Mod[T[[k]] * T[[j]], 11], {k, 1, Length[K]}, {j, 1, Length[K]}];

M = TableForm[L, TableHeadings → {K, K}];

12 = Table[Insert[L[[i]], K[[i]], 1], {i, Length[L]}];

group1 = Insert[K, "*", 1];

13 = Insert[12, group1, 1];

DisplayForm[FrameBox[GridBox[13, RowLines → True, ColumnLines → True]]]
```

*	0	1	2	3	4	5	6	7	8	9	10
0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7	8	9	10
2	0	2	4	6	8	10	1	3	5	7	9
3	0	3	6	9	1	4	7	10	2	5	8
4	0	4	8	1	5	9	2	6	10	3	7
5	0	5	10	4	9	3	8	2	7	1	6
6	0	6	1	7	2	8	3	9	4	10	5
7	0	7	3	10	6	2	9	5	1	8	4
8	0	8	5	2	10	7	4	1	9	6	3
9	0	9	7	5	3	1	10	8	6	4	2
10	0	10	9	8	7	6	5	4	3	2	1

Appendix 2

The *Mathematica* code for the addition table and the encryption and decryption operations using the elliptic curve $y^2 = x^3 + 5x + 4$ over Z_{11}.

```
Off[General::spell]
Off[General::spell1]
```

Defining the RHS of the elliptic curve equation over Z_p, when p>3 is prime

```
f[a_, b_, x_, p_] := Mod[x^3 + (a * x) + b, p]
```

Getting the point possibilities on an elliptic curve over Z_{11}

```
L = Partition[Flatten[Table[{i, j}, {i, 0, 10}, {j, 0, 10}]], 2]
```

```
{{0, 0}, {0, 1}, {0, 2}, {0, 3}, {0, 4}, {0, 5}, {0, 6}, {0, 7}, {0, 8}, {0, 9}, {0, 10},
 {1, 0}, {1, 1}, {1, 2}, {1, 3}, {1, 4}, {1, 5}, {1, 6}, {1, 7}, {1, 8}, {1, 9}, {1, 10},
 {2, 0}, {2, 1}, {2, 2}, {2, 3}, {2, 4}, {2, 5}, {2, 6}, {2, 7}, {2, 8}, {2, 9},
 {2, 10}, {3, 0}, {3, 1}, {3, 2}, {3, 3}, {3, 4}, {3, 5}, {3, 6}, {3, 7}, {3, 8},
 {3, 9}, {3, 10}, {4, 0}, {4, 1}, {4, 2}, {4, 3}, {4, 4}, {4, 5}, {4, 6}, {4, 7},
 {4, 8}, {4, 9}, {4, 10}, {5, 0}, {5, 1}, {5, 2}, {5, 3}, {5, 4}, {5, 5}, {5, 6},
 {5, 7}, {5, 8}, {5, 9}, {5, 10}, {6, 0}, {6, 1}, {6, 2}, {6, 3}, {6, 4}, {6, 5},
 {6, 6}, {6, 7}, {6, 8}, {6, 9}, {6, 10}, {7, 0}, {7, 1}, {7, 2}, {7, 3}, {7, 4},
 {7, 5}, {7, 6}, {7, 7}, {7, 8}, {7, 9}, {7, 10}, {8, 0}, {8, 1}, {8, 2}, {8, 3},
 {8, 4}, {8, 5}, {8, 6}, {8, 7}, {8, 8}, {8, 9}, {8, 10}, {9, 0}, {9, 1}, {9, 2},
 {9, 3}, {9, 4}, {9, 5}, {9, 6}, {9, 7}, {9, 8}, {9, 9}, {9, 10}, {10, 0}, {10, 1},
 {10, 2}, {10, 3}, {10, 4}, {10, 5}, {10, 6}, {10, 7}, {10, 8}, {10, 9}, {10, 10}}
```

Testing which of the point possibilities lie on the elliptic curve $y^2 = x^3 + 5x + 4$ over Z_{11}

```
A = Table[Mod[L[[m, 2]]^2, 11] == f[5, 4, L[[m, 1]], 11], {m, 1, Length[L]}]
```

```
{False, False, True, False, False, False, False, False, False, True, False, False, False,
 False, False, False, False, False, False, False, False, False, True, False, False,
 False, False, False, False, False, False, False, False, False, False, False, False,
 False, False, False, False, False, False, False, True, False, False, False, False,
 False, False, False, False, False, False, True, False, False, False, False, False,
 False, False, False, False, False, False, False, False, False, False, False, False,
 False, False, False, False, False, False, False, False, False, False, False, False,
 False, False, False, False, False, False, False, False, False, False, False, False,
 False, False, False, False, False, False, False, False, False, False, False, False,
 False, False, False, False, True, False, False, False, False, True, False, False}
```

```
P = Position[A, True]
```

```
{{3}, {10}, {23}, {45}, {56}, {114}, {119}}
```

```
Q = Partition[Flatten[Table[L[[P[[n]]]], {n, 1, Length[P]}]], 2]
```

```
{{0, 2}, {0, 9}, {2, 0}, {4, 0}, {5, 0}, {10, 3}, {10, 8}}
```

Adding in the point at infinity, $\{\infty , \infty\}$

```
Z = {Q, ∞}
```

```
{{{0, 2}, {0, 9}, {2, 0}, {4, 0}, {5, 0}, {10, 3}, {10, 8}}, ∞}
```

The points that lie on the elliptic curve $y^2 = x^3 + 5x + 4$ over Z_{11}

```
U = Partition[Flatten[{Z, ∞}], 2]
```

```
{{0, 2}, {0, 9}, {2, 0}, {4, 0}, {5, 0}, {10, 3}, {10, 8}, {∞, ∞}}
```

Defining the addition operation for elliptic curves over Z_p, when $p > 3$ is prime

```
AddFunction[xP_, yP_, xQ_, yQ_, a_, p_] :=
 If[xP == ∞ && yP == ∞, {xQ, yQ}, If[xQ == ∞ && yQ == ∞, {xP, yP}, If[(xP == xQ && yP == Mod[-yQ, p]),
    {∞, ∞}, If[xP == xQ && yP == yQ, {Mod[(Mod[3 xP^2 + a, p] * PowerMod[2 yP, -1, p])^2 - 2 xP, p],
      Mod[-yP + (Mod[3 xP^2 + a, p] * PowerMod[2 yP, -1, p])
          (xP - (Mod[(Mod[3 xP^2 + a, p] * PowerMod[2 yP, -1, p])^2 - 2 xP, p])), p]},
     {Mod[(Mod[yP - yQ, p] * PowerMod[xP - xQ, -1, p])^2 - xP - xQ, p],
      Mod[-yP + (Mod[yP - yQ, p] * PowerMod[xP - xQ, -1, p])
          (xP - (Mod[(Mod[yP - yQ, p] * PowerMod[xP - xQ, -1, p])^2 - xP - xQ, p])), p]}]]]]
```

Computing the addition table for the points on the elliptic curve $y^2 = x^3 + 5x + 4$ over Z_{11}

```
T = Table[AddFunction[U[[m, 1]], U[[m, 2]], U[[1, 1]], U[[1, 2]], 5, 11],
   {m, 1, Length[U]}, {1, 1, Length[U]}];
```

Addition table for the points on the elliptic curve $y^2 = x^3 + 5x + 4$ over Z_{11}

```
L2 = Table[Insert[T[[j]], U[[j]], 1], {j, 1, Length[T]}];

group1 = Insert[U, "*", 1];

L3 = Insert[L2, group1, 1] /. {∞, ∞} → ∞;

DisplayForm[
 FrameBox[GridBox[L3, RowLines → True, ColumnLines → True, ColumnSpacings → .01]]]
```

*	{0, 2}	{0, 9}	{2, 0}	{4, 0}	{5, 0}	{10, 3}	{10, 8}	∞
{0, 2}	{5, 0}	∞	{10, 8}	{10, 3}	{0, 9}	{2, 0}	{4, 0}	{0, 2}
{0, 9}	∞	{5, 0}	{10, 3}	{10, 8}	{0, 2}	{4, 0}	{2, 0}	{0, 9}
{2, 0}	{10, 8}	{10, 3}	∞	{5, 0}	{4, 0}	{0, 9}	{0, 2}	{2, 0}
{4, 0}	{10, 3}	{10, 8}	{5, 0}	∞	{2, 0}	{0, 2}	{0, 9}	{4, 0}
{5, 0}	{0, 9}	{0, 2}	{4, 0}	{2, 0}	∞	{10, 8}	{10, 3}	{5, 0}
{10, 3}	{2, 0}	{4, 0}	{0, 9}	{0, 2}	{10, 8}	{5, 0}	∞	{10, 3}
{10, 8}	{4, 0}	{2, 0}	{0, 2}	{0, 9}	{10, 3}	∞	{5, 0}	{10, 8}
∞	{0, 2}	{0, 9}	{2, 0}	{4, 0}	{5, 0}	{10, 3}	{10, 8}	∞

Computing β in the encryption operation

```
β = Function[{xα, yα, a, p, z}, For[i = 1; S = AddFunction[∞, ∞, xα, yα, a, p],
    i < z, i++, S = AddFunction[S[[1]], S[[2]], xα, yα, a, p]]; S][10, 3, 5, 11, 3]
```

```
{10, 8}
```

Computing y_1=kα=2(10,3) in the encryption operation

```
y1 = Function[{xα, yα, a, p, k}, For[i = 1; S = AddFunction[∞, ∞, xα, yα, a, p],
    i < k, i++, S = AddFunction[S[[1]], S[[2]], xα, yα, a, p]]; S][10, 3, 5, 11, 2]
```

```
{5, 0}
```

Computing $k\beta$=2(10,8) in the encryption operation

```
kβ = Function[{xβ, yβ, a, p, k}, For[i = 1; S = AddFunction[∞, ∞, xβ, yβ, a, p], i < k, i++,
    S = AddFunction[S[[1]], S[[2]], xβ, yβ, a, p]]; S][β[[1]], β[[2]], 5, 11, 2]
```

```
{5, 0}
```

Computing $y_2 = x + k\beta = (2, 0) + (5, 0)$ in the encryption operation

```
y2 = AddFunction[2, 0, kβ[[1]], kβ[[2]], 5, 11]
```

```
{4, 0}
```

The encrypted message (y_1, y_2)

```
Code = {y1, y2}
```

```
{{5, 0}, {4, 0}}
```

Computing $zy_1 = 3\,(5, 0)$ in the decryption operation

```
Decrypt =
 Function[{xD, yD, a, p, z}, For[i = 1; S = AddFunction[∞, ∞, xD, yD, a, p], i < z, i++,
    S = AddFunction[S[[1]], S[[2]], xD, yD, a, p]]; S][Code[[1, 1]], Code[[1, 2]], 5, 11, 3]
```

```
{5, 0}
```

Computing $x = y_2 - zy_1 = (4, 0) - (5, 0)$ in the decryption operation

```
Decryption = AddFunction[Code[[2, 1]], Code[[2, 2]], Decrypt[[1]], -Decrypt[[2]], 5, 11]
```

```
{2, 0}
```

Appendix 3

The *Mathematica* code for the addition table and the encryption and decryption operations using the elliptic curve $y^2 + xy = x^3 + (x + 1)x^2 + (x + 1)$ over F_2^3.

```
Off[General::spell]
Off[General::spell1]

<< Algebra`PolynomialPowerMod`
```

Defining the RHS of the elliptic curve equation over F_{2^m}

```
f[a_, b_, h_] := h^3 + (a * h^2) + b
```

Defining the irreducible polynomial, p

```
g[x_] := x^3 + x + 1
```

Getting the coefficient possibilities for an elliptic curve over F_{2^3}

```
Coefficients = Partition[Flatten[Table[{i, j, k}, {i, 0, 1}, {j, 0, 1}, {k, 0, 1}]], 3]
```

```
{{0, 0, 0}, {0, 0, 1}, {0, 1, 0}, {0, 1, 1}, {1, 0, 0}, {1, 0, 1}, {1, 1, 0}, {1, 1, 1}}
```

Converting these sets of coefficients into their corresponding polynomial representations

```
F = Table[Sum[Coefficients[[i]][[j]] x^(3-j), {j, 1, 3}], {i, 1, Length[Coefficients]}]
```

```
{0, 1, x, 1 + x, x^2, 1 + x^2, x + x^2, 1 + x + x^2}
```

Getting the point possibilities for an elliptic curve over F_{2^3}

```
L = Partition[Flatten[Table[{F[[i]], F[[j]]}, {i, 1, Length[F]}, {j, 1, Length[F]}]], 2]
```

```
{{0, 0}, {0, 1}, {0, x}, {0, 1 + x}, {0, x^2}, {0, 1 + x^2}, {0, x + x^2}, {0, 1 + x + x^2},
 {1, 0}, {1, 1}, {1, x}, {1, 1 + x}, {1, x^2}, {1, 1 + x^2}, {1, x + x^2}, {1, 1 + x + x^2},
 {x, 0}, {x, 1}, {x, x}, {x, 1 + x}, {x, x^2}, {x, 1 + x^2}, {x, x + x^2}, {x, 1 + x + x^2},
 {1 + x, 0}, {1 + x, 1}, {1 + x, x}, {1 + x, 1 + x}, {1 + x, x^2}, {1 + x, 1 + x^2}, {1 + x, x + x^2},
 {1 + x, 1 + x + x^2}, {x^2, 0}, {x^2, 1}, {x^2, x}, {x^2, 1 + x}, {x^2, x^2}, {x^2, 1 + x^2},
 {x^2, x + x^2}, {x^2, 1 + x + x^2}, {1 + x^2, 0}, {1 + x^2, 1}, {1 + x^2, x}, {1 + x^2, 1 + x},
 {1 + x^2, x^2}, {1 + x^2, 1 + x^2}, {1 + x^2, x + x^2}, {1 + x^2, 1 + x + x^2}, {x + x^2, 0},
 {x + x^2, 1}, {x + x^2, x}, {x + x^2, 1 + x}, {x + x^2, x^2}, {x + x^2, 1 + x^2}, {x + x^2, x + x^2},
 {x + x^2, 1 + x + x^2}, {1 + x + x^2, 0}, {1 + x + x^2, 1}, {1 + x + x^2, x}, {1 + x + x^2, 1 + x},
 {1 + x + x^2, x^2}, {1 + x + x^2, 1 + x^2}, {1 + x + x^2, x + x^2}, {1 + x + x^2, 1 + x + x^2}}
```

Testing which of the point possibilities lie on the elliptic curve $y^2 + xy = x^3 + (x+1)x^2 + (x+1)$ over F_{2^3}

```
M = Table[PolynomialMod[(L[[i, 2]]^2 + (L[[i, 1]] * L[[i, 2]])), 2], {i, 1, Length[L]}];

R = Table[PolynomialMod[M[[i]], {g[x], 2}], {i, 1, Length[L]}];

S = Table[PolynomialMod[f[1 + x, 1 + x, L[[i, 1]]], 2], {i, 1, Length[L]}];

T = Table[PolynomialMod[S[[i]], {g[x], 2}], {i, 1, Length[L]}];
```

```
J = Table[R[[i]] === T[[i]], {i, 1, Length[L]}]
```

```
{False, False, False, False, False, False, False, True, False, False, False, False, False,
 False, False, False, False, False, False, False, False, False, False, False, False, False,
 False, False, False, False, True, True, False, False, False, True, False, False, False,
 True, False, False, False, False, False, False, False, False, False, False, False, False,
 False, False, False, False, False, False, False, False, False, False, False, False, False}
```

```
Pos = Position[J, True]
```

```
{{8}, {30}, {31}, {35}, {39}}
```

```
Z = Partition[Flatten[Table[L[[Pos[[n]]]], {n, 1, Length[Pos]}]], 2]
```

$$\{\{0,\ 1+x+x^2\},\ \{1+x,\ 1+x^2\},\ \{1+x,\ x+x^2\},\ \{x^2,\ x\},\ \{x^2,\ x+x^2\}\}$$

Adding in the point at infinity, $\{\infty, \infty\}$

```
A = {Z, ∞}
```

$$\{\{\{0,\ 1+x+x^2\},\ \{1+x,\ 1+x^2\},\ \{1+x,\ x+x^2\},\ \{x^2,\ x\},\ \{x^2,\ x+x^2\}\},\ \infty\}$$

The points that lie on the elliptic curve $y^2 + xy = x^3 + (x+1)x^2 + (x+1)$ over F_{2^3}

```
U = Partition[Flatten[{A, ∞}], 2]
```

```
{{0, 1 + x + x^2}, {1 + x, 1 + x^2}, {1 + x, x + x^2}, {x^2, x}, {x^2, x + x^2}, {∞, ∞}}
```

Defining the addition operation for elliptic curves over F_{2^m}

```
AddFunction[xP_, yP_, xQ_, yQ_, a_, p_] :=
 If[xP === ∞ && yP === ∞, {xQ, yQ}, If[xQ === ∞ && yQ === ∞, {xP, yP},
   If[xP === xQ && yQ === PolynomialMod[xP + yP, 2], {∞, ∞}, If[xP === 0 && xQ === 0 && yP === yQ,
     {∞, ∞}, If[xP === xQ && yP === yQ, {PolynomialMod[(PolynomialMod[xP, 2] +
            (PolynomialMod[yP, 2] * PolynomialPowerMod[xP, -1, {p, 2}]))^2 + (PolynomialMod[
           xP, 2] + (PolynomialMod[yP, 2] * PolynomialPowerMod[xP, -1, {p, 2}])) + a,
       {p, 2}], PolynomialMod[xP^2 + ((PolynomialMod[xP, 2] + (PolynomialMod[yP, 2] *
              PolynomialPowerMod[xP, -1, {p, 2}])) + 1) * (PolynomialMod[(PolynomialMod[
             xP, 2] + (PolynomialMod[yP, 2] * PolynomialPowerMod[xP, -1, {p, 2}]))^2 +
           (PolynomialMod[xP, 2] + (PolynomialMod[yP, 2] * PolynomialPowerMod[
               xP, -1, {p, 2}])) + a, {p, 2}]), {p, 2}]}, {PolynomialMod[
       (PolynomialMod[(yP - yQ), 2] * PolynomialPowerMod[(xP + xQ), -1, {p, 2}])^2 +
        (PolynomialMod[(yP - yQ), 2] * PolynomialPowerMod[(xP + xQ), -1, {p, 2}]) +
        xP + xQ + a, {p, 2}], PolynomialMod[
       (PolynomialMod[(yP - yQ), 2] * PolynomialPowerMod[(xP + xQ), -1, {p, 2}])
         (xP + (PolynomialMod[(PolynomialMod[(yP - yQ), 2] * PolynomialPowerMod[(xP + xQ),
              -1, {p, 2}])^2 + (PolynomialMod[(yP - yQ), 2] * PolynomialPowerMod[
             (xP + xQ), -1, {p, 2}]) + xP + xQ + a, {p, 2}])) + (PolynomialMod[
         (PolynomialMod[(yP - yQ), 2] * PolynomialPowerMod[(xP + xQ), -1, {p, 2}])^2 +
          (PolynomialMod[(yP - yQ), 2] * PolynomialPowerMod[(xP + xQ), -1, {p, 2}]) +
          xP + xQ + a, {p, 2}]) + yP, {p, 2}]}]]]]]]
```

Computing the addition table for the points on the elliptic curve $y^2 + xy = x^3 + (x+1)x^2 + (x+1)$ over F_{2^3}

```
Add = Table[AddFunction[U[[m, 1]], U[[m, 2]], U[[1, 1]], U[[1, 2]], x + 1, x^3 + x + 1],
   {m, 1, Length[U]}, {1, 1, Length[U]}];
```

Addition table for the points that lie on the elliptic curve $y^2 + xy = x^3 + (x+1)x^2 + (x+1)$ over F_{2^3}

```
L2 = Table[Insert[Add[[j]], U[[j]], 1], {j, 1, Length[Add]}];

group1 = Insert[U, "*", 1];

L3 = Insert[L2, group1, 1] /. {∞, ∞} → ∞;
```

```
DisplayForm[
 FrameBox[GridBox[L3, RowLines → True, ColumnLines → True, ColumnSpacings → .01]]]
```

*	$\{0, 1+x+x^2\}$	$\{1+x, 1+x^2\}$	$\{1+x, x+x^2\}$	$\{x^2, x\}$	$\{x^2, x+x^2\}$	∞
$\{0, 1+x+x^2\}$	∞	$\{x^2, x\}$	$\{x^2, x+x^2\}$	$\{1+x, 1+x^2\}$	$\{1+x, x+x^2\}$	$\{0, 1+x+x^2\}$
$\{1+x, 1+x^2\}$	$\{x^2, x\}$	$\{1+x, x+x^2\}$	∞	$\{x^2, x+x^2\}$	$\{0, 1+x+x^2\}$	$\{1+x, 1+x^2\}$
$\{1+x, x+x^2\}$	$\{x^2, x+x^2\}$	∞	$\{1+x, 1+x^2\}$	$\{0, 1+x+x^2\}$	$\{x^2, x\}$	$\{1+x, x+x^2\}$
$\{x^2, x\}$	$\{1+x, 1+x^2\}$	$\{x^2, x+x^2\}$	$\{0, 1+x+x^2\}$	$\{1+x, x+x^2\}$	∞	$\{x^2, x\}$
$\{x^2, x+x^2\}$	$\{1+x, x+x^2\}$	$\{0, 1+x+x^2\}$	$\{x^2, x\}$	∞	$\{1+x, 1+x^2\}$	$\{x^2, x+x^2\}$
∞	$\{0, 1+x+x^2\}$	$\{1+x, 1+x^2\}$	$\{1+x, x+x^2\}$	$\{x^2, x\}$	$\{x^2, x+x^2\}$	∞

Computing $\beta = z\alpha = 2\,(x^2, x)$ in the encryption operation

```
β = Function[{xα, yα, a, p, z}, For[i = 1; Y = AddFunction[∞, ∞, xα, yα, a, p], i < z, i++,
    Y = AddFunction[Y[[1]], Y[[2]], xα, yα, a, p]]; Y][x^2, x, x + 1, x^3 + x + 1, 2]
```

```
{1 + x, x + x^2}
```

Computing $y_1 = k\alpha = 4\,(x^2, x)$ in the encryption operation

```
y1 = Function[{xα, yα, a, p, k}, For[i = 1; Y = AddFunction[∞, ∞, xα, yα, a, p], i < k,
    i++, Y = AddFunction[Y[[1]], Y[[2]], xα, yα, a, p]]; Y][x^2, x, x + 1, x^3 + x + 1, 4]
```

```
{1 + x, 1 + x^2}
```

Computing $k\beta = 4\,(1+x, x+x^2)$ in the encryption operation

```
kβ = Function[{xβ, yβ, a, p, k}, For[i = 1; Y = AddFunction[∞, ∞, xβ, yβ, a, p], i < k, i++,
    Y = AddFunction[Y[[1]], Y[[2]], xβ, yβ, a, p]]; Y][β[[1]], β[[2]], x + 1, x^3 + x + 1, 4]
```

```
{1 + x, x + x^2}
```

Computing $y_2 = x + k\beta = (x^2, x) + (1+x, x+x^2)$ in the encryption operation

```
y2 = AddFunction[x^2, x, kβ[[1]], kβ[[2]], x + 1, x^3 + x + 1]
```

```
{0, 1 + x + x^2}
```

The encrypted message (y_1, y_2)

```
Code = {y1, y2}
```

```
{{1 + x, 1 + x^2}, {0, 1 + x + x^2}}
```

Computing $zy_1 = 2\,(1+x, 1+x^2)$ in the decryption operation

```
Decrypt = Function[{xD, yD, a, p, z}, For[i = 1; Y = AddFunction[∞, ∞, xD, yD, a, p],
    i < z, i++, Y = AddFunction[Y[[1]], Y[[2]], xD, yD, a, p]]; Y][
  Code[[1, 1]], Code[[1, 2]], x + 1, x^3 + x + 1, 2]
```

$\{1 + x,\ x + x^2\}$

Computing $x = y_2 - zy_1 = (0,\ 1 + x + x^2) - (1 + x,\ x + x^2)$ in the decryption operation

```
Decryption = AddFunction[Code[[2, 1]], Code[[2, 2]],
  Decrypt[[1]], (Decrypt[[1]] + Decrypt[[2]]), x + 1, x^3 + x + 1]
```

$\{x^2,\ x\}$